



Requêtes sur Base de Données Géographique Voirie et Transports Collectifs

RAPPORT de PHASE 1

30/12/2008

Mises à jour du rapport

Numéro de Version	Date	Auteur principal	Résumé des modifications
1	26/11/2008	L. Dezou (MobiGIS)	Création
2	30/12/2008	L. Dezou (MobiGIS)	Intégration des remarques de P. Gendre (CETE)

Contacts

Organisme/Société	Nom du correspondant/Adresse/Téléphone/E-mail
CETE Méditerranée	Nom du correspondant : Patrick Gendre Adresse: Avenue Albert Einstein CS 70499, 13593 Aix-en-Provence Cedex 3 Téléphone : 04 42 24 76 87 E-mail : pat.gendre@developpement-durable.gouv.fr
MobiGIS	Nom du correspondant : Laurent Dezou Adresse: Rue du Lanoux 31330 Grenade Téléphone : 05 81 60 80 82 E-mail : ldezou@mobigis.fr


MobiGIS


Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE
CETE
Méditerranée

Sommaire

I. CONTEXTE

Le CETE Méditerranée contribue au programme PREDIM, à la normalisation de l'information concernant les Transports Collectifs (TC), et a produit notamment en 2006 un rapport sur l'utilisation des Systèmes d'Information Géographique (SIG) pour l'analyse des réseaux piétons et cyclables (modes doux, http://www.cete-mediterranee.fr/tt13/www/article.php3?id_article=101).

Le CETE participe par ailleurs au projet POTIMART visant à développer des solutions SIG open source pour l'information multimodale (www.potimart.org et http://www.cete-mediterranee.fr/tt13/www/article.php3?id_article=118).

Le projet *Requêtes BD SIG IMM* vise à développer des outils open source permettant quelques analyses simples sur une base de données routière et TC :

- « perméabilité » des modes doux (marche à pied, vélo) sur un réseau routier,
- fréquence et vitesse des lignes sur un réseau TC décrit dans une base de données Chouette.

Dans le cadre de la phase 1 de la prestation de la société MobiGIS, des requêtes SQL sont mises au point. Elles permettent :

- D'ajouter la composante spatiale aux données TC de type Chouette,
- D'intégrer les données voirie BD Topo dans la base PostgreSQL/PostGIS,
- De calculer des indicateurs prédéfinis.

A la suite de ces traitements, les résultats peuvent être cartographiés grâce à l'utilisation du logiciel SIG libre Quantum GIS (QGIS). Les requêtes seront testées et démontrées sur des données réelles existantes fournies par le CETE pour un site pilote (Toulouse).

La phase 2 de la prestation consistera à « packager » les requêtes développées dans le cadre de la phase 1 et à produire un rapport de perspectives.

II. OBJET DU DOCUMENT

Le rapport phase 1 du projet *Requête BD SIG IMM* a pour objet :

- De lister les constituants de la solution et décrire leur installation,
- De décrire les indicateurs et les calculs opérés,
- De fournir le mode d'utilisation des scripts.

Le rapport phase 2 proposera des perspectives d'extension et de mise en œuvre.

Le présent rapport (phase 1) est un rapport technique qui décrit le travail réalisé, destiné essentiellement au CETE. Le rapport de phase 2, qui présentera les données voirie/TC utilisées et inclura un mode d'emploi des requêtes, a vocation à être diffusé de manière large aux utilisateurs potentiels.





III. DOCUMENT APPLICABLE ET RÉFÉRENCE

[DA1] Cahier des charges Requêtes BD SIG SIG IMM, version 0.4 du 10/09/08

Consultation en vue d'une Prestation Intellectuelle

IV. GLOSSAIRE

BD	Base de Données
IMM	Information Multi Modale
QGIS	Quantum GIS Open Source
PREDIM	Plate-forme de Recherche et d'Expérimentation pour le Développement de l'Information Multimodale
SIG	Système d'Information Géographique

V. LISTE DES CONSTITUANTS

Liste des constituants et version utilisées :

Python	http://www.python.org	2.5.2
PostGreSQL	http://www.postgresql.org/	8.4.4
PostGIS	http://postgis.refractions.net	1.3.3
QuantumGIS ou QGIS	http://www.qgis.org	0.11
SPIT	Outil d'importation de shapefile vers PostGIS	Extension de QGIS

VI. INSTALLATION

A. Python

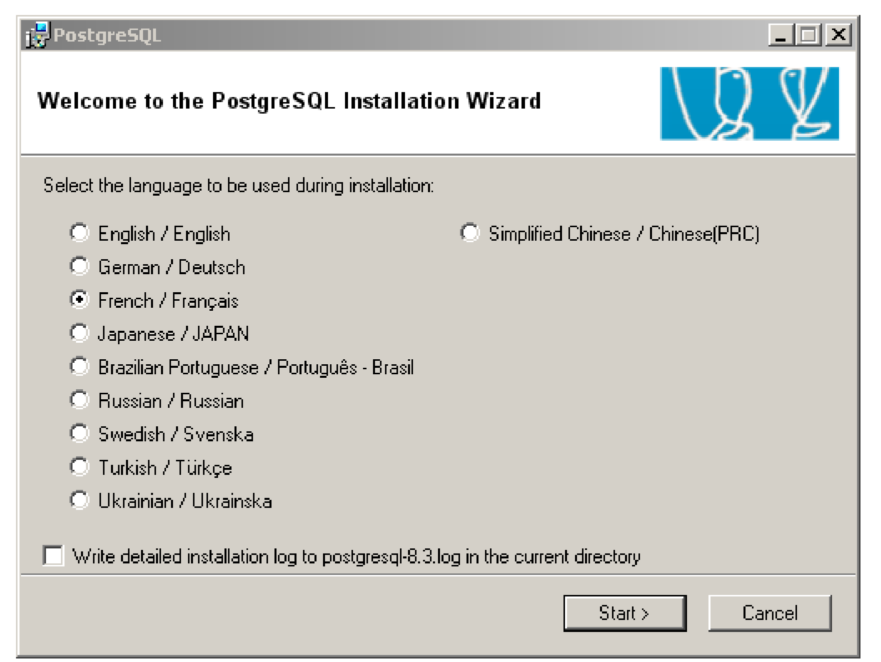
Pour utiliser le langage Pl/Python.

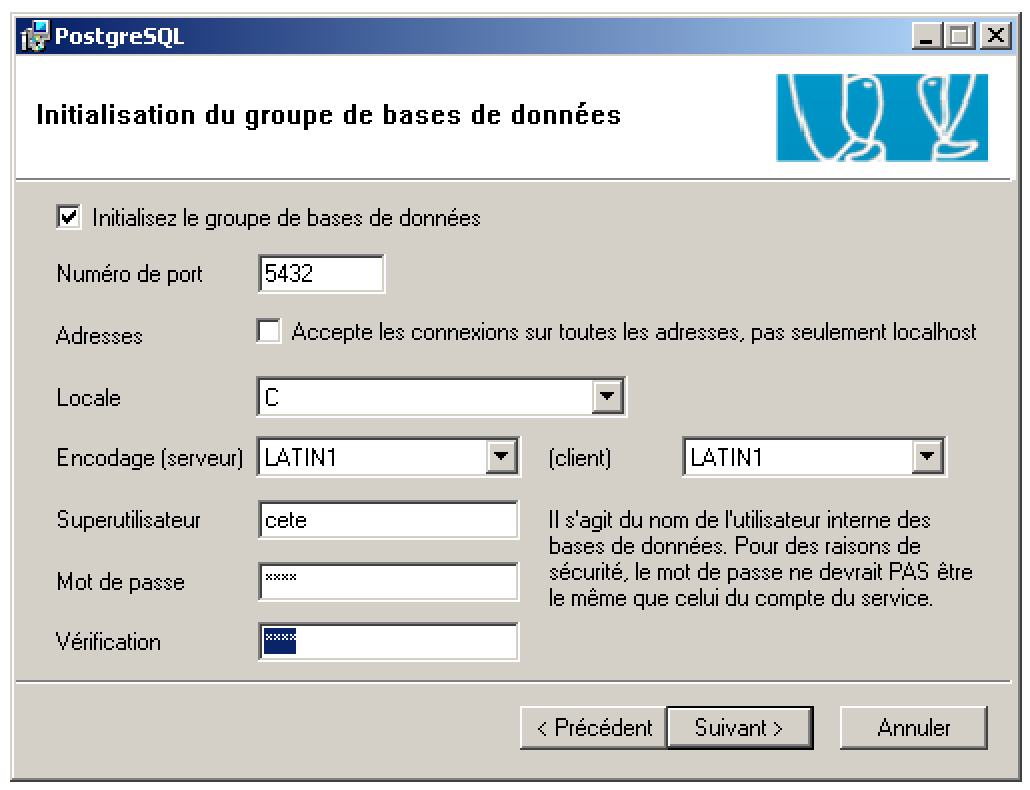
Installation à partir de python-2.5.2.msi

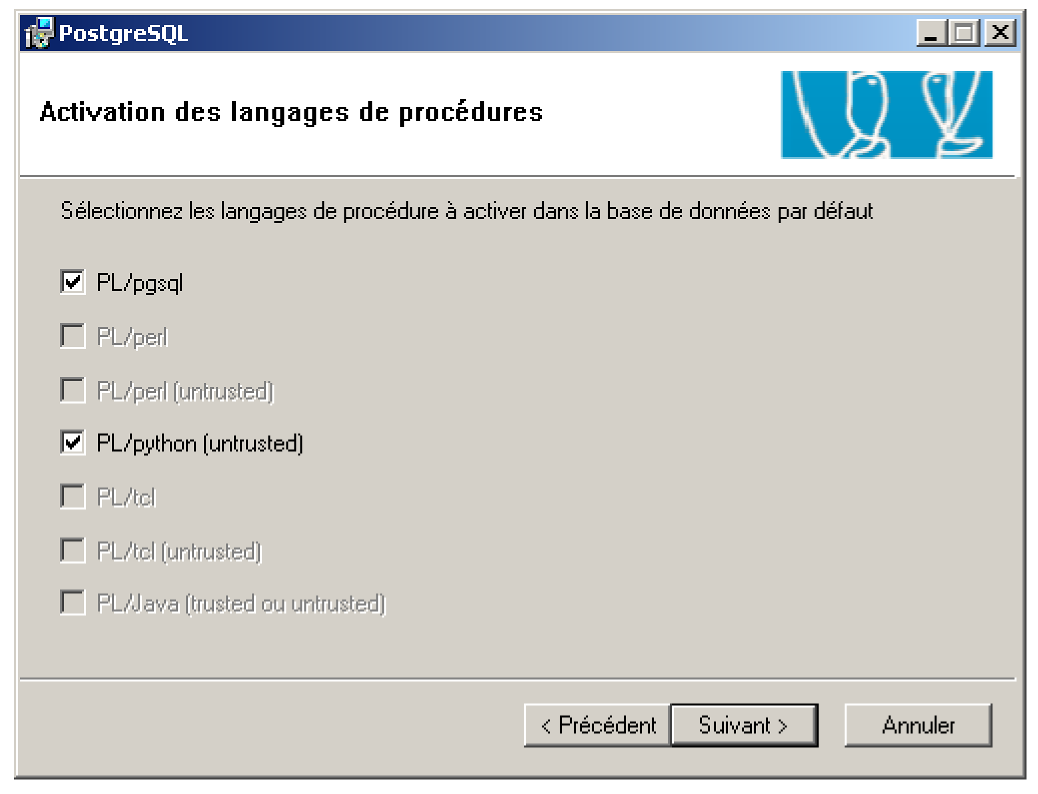
B. PostGreSQL

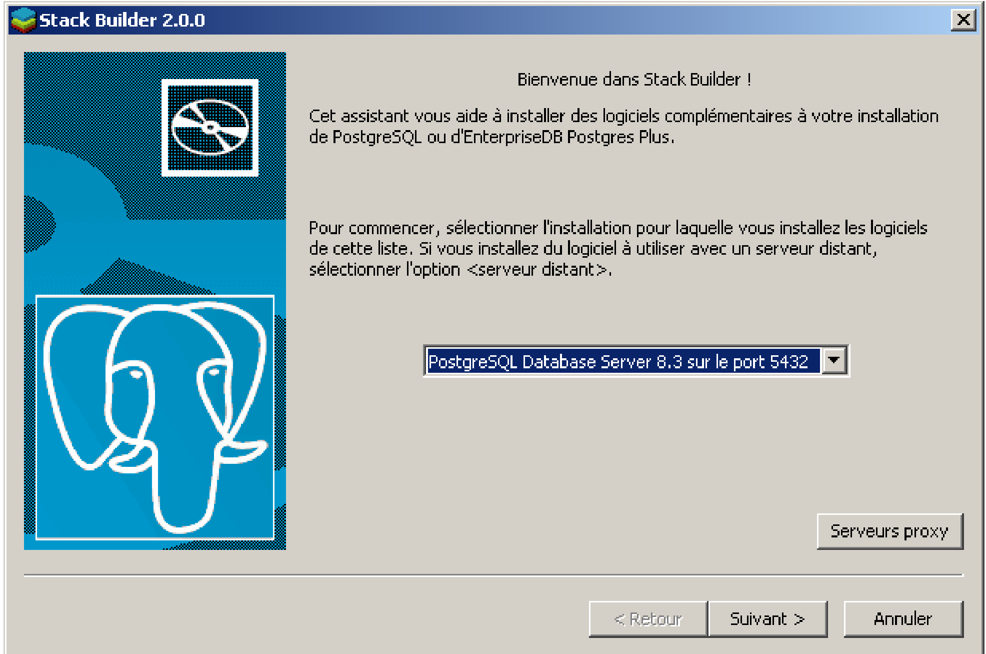
Manuel utilisateur : http://www.postgresql.org/docs/8.3/static/index.html

Installation à partir de postgresql-8.3.msi


PostgreSQL
Welcome to the PostgreSQL Installation Wizard
Select the language to be used during installation:
English / English
Simplified Chinese / Chinese(PRC)
German / Deutsch
French / Français
Japanese / JAPAN
Brazilian Portuguese / Português - Brasil
Russian / Russian
Swedish / Svenska
Turkish / Türkçe
Ukrainian / Ukrainska
Write detailed installation log to postgresql-8.3.log in the current directory
Start >
Cancel

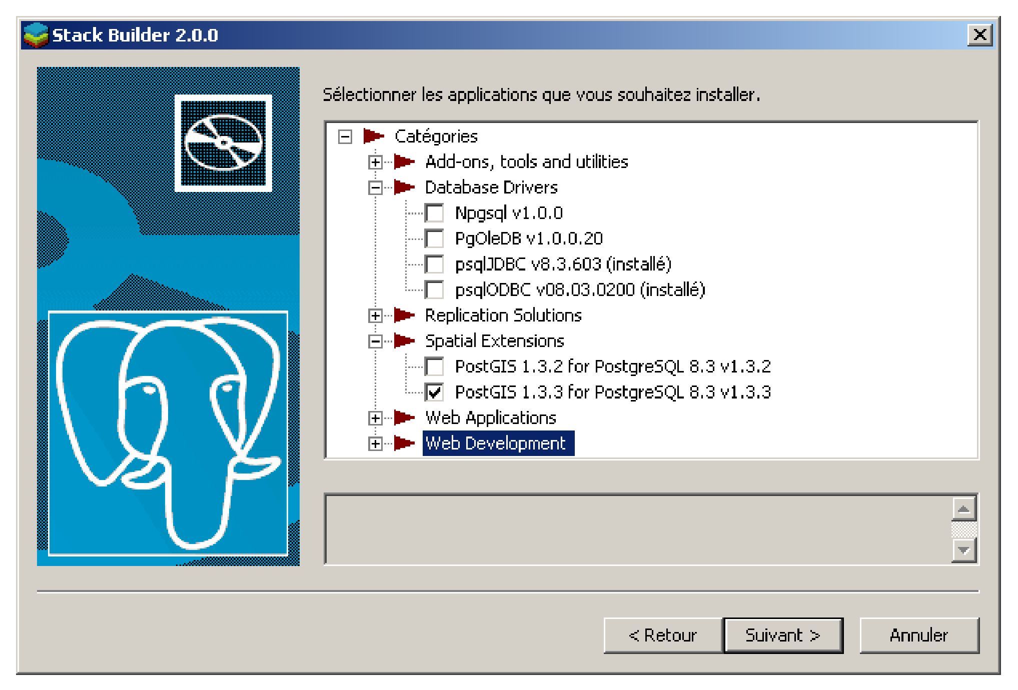

PostgreSQL
Initialisation du groupe de bases de données
Initialisez le groupe de bases de données
Numéro de port
5432
Adresses
Accepte les connexions sur toutes les adresses, pas seulement localhost
Locale
C
Encodage (serveur)
LATIN1
(client)
LATIN1
Superutilisateur
cete
Il s'agit du nom de l'utilisateur interne des bases de données. Pour des raisons de sécurité, le mot de passe ne devrait PAS être le même que celui du compte du service.
Mot de passe
Vérification
< Précédent
Suivant >
Annuler





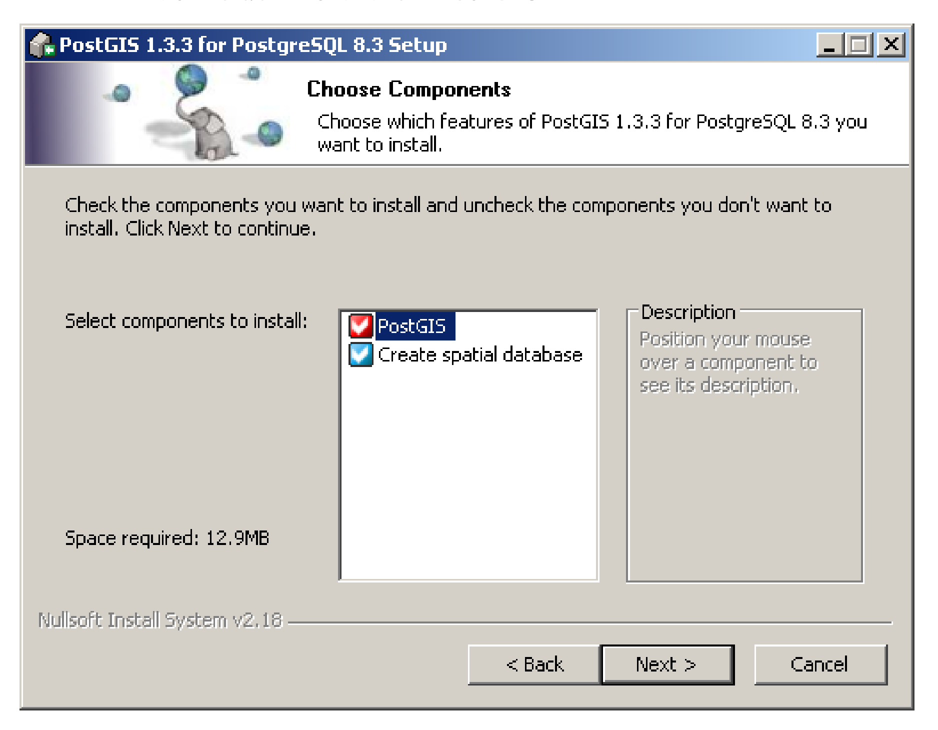
C. <u>PostGIS</u>

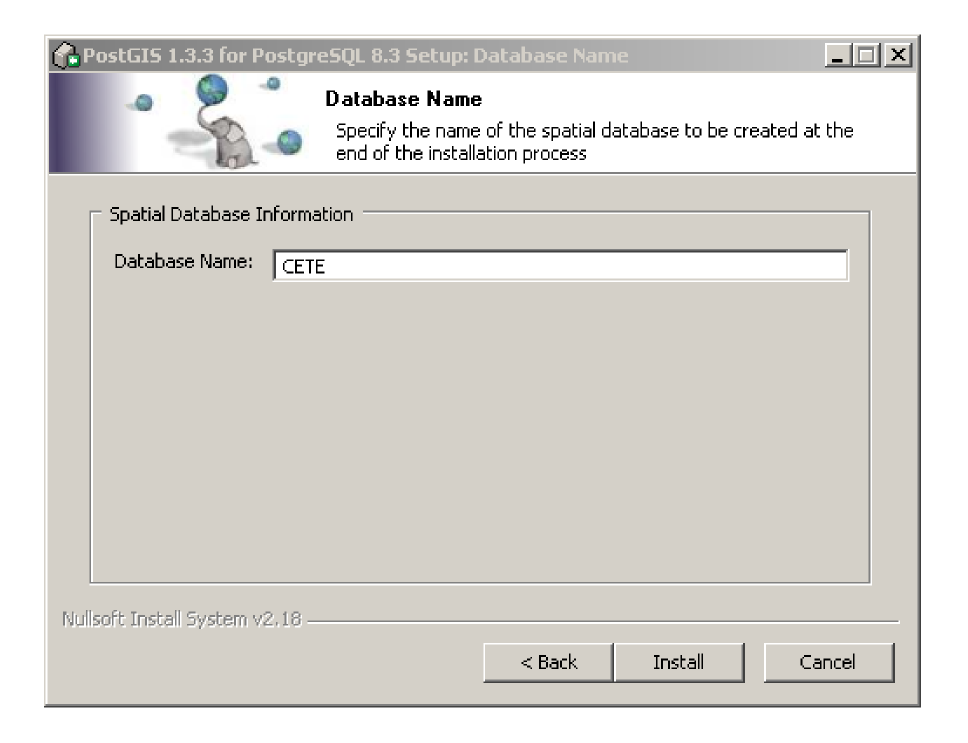
Installation à l'aide du Stack Builder PostgreSQL (Constructeur de la pile applicative) :


Stack Builder 2.0.0
Sélectionner les applications que vous souhaitez installer.
Catégories
Add-ons, tools and utilities
Database Drivers
Npgsql v1.0.0
PgOleDB v1.0.0.20
psqlJDBC v8.3.603 (installé)
psqlODBC v08.03.0200 (installé)
Replication Solutions
Spatial Extensions
PostGIS 1.3.2 for PostgreSQL 8.3 v1.3.2
PostGIS 1.3.3 for PostgreSQL 8.3 v1.3.3
Web Applications
Web Development
< Retour
Suivant >
Annuler

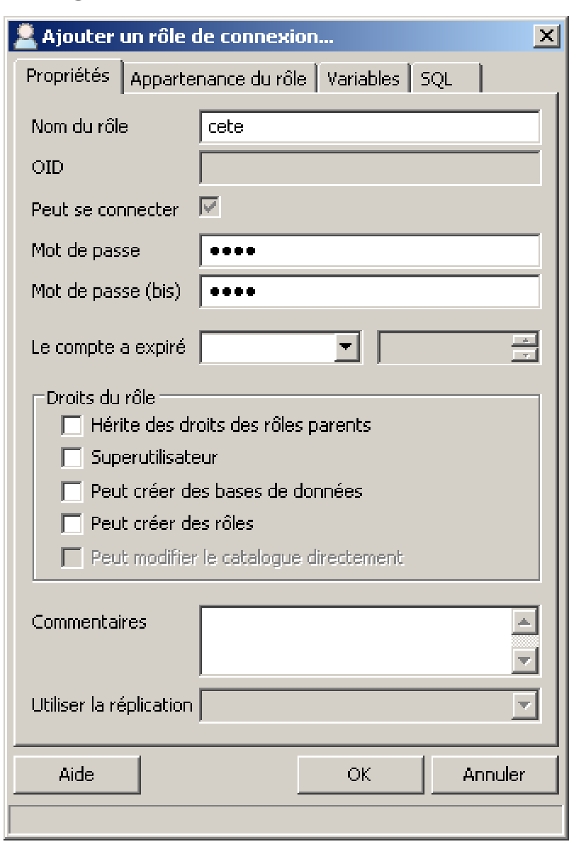
Création de la base PostGIS, deux possibilités :

- A la suite de l'installation PostGIS

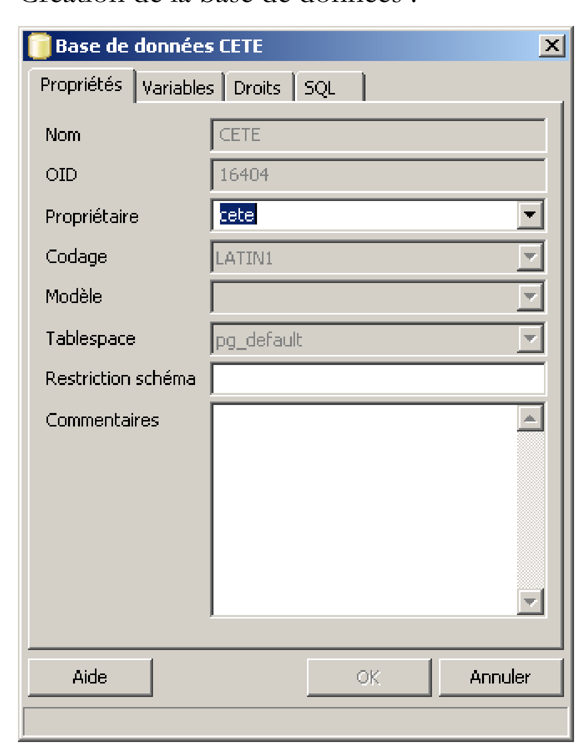




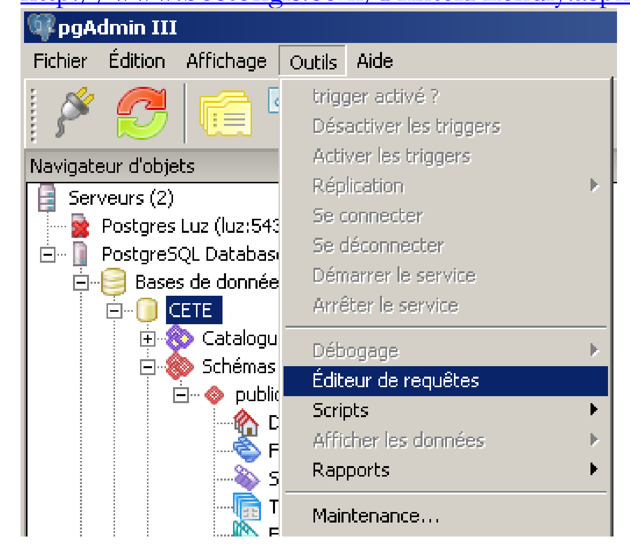
- En dehors de l'installation PostGIS :
  - Via PgAdmin ; création de l'utilisateur cete :



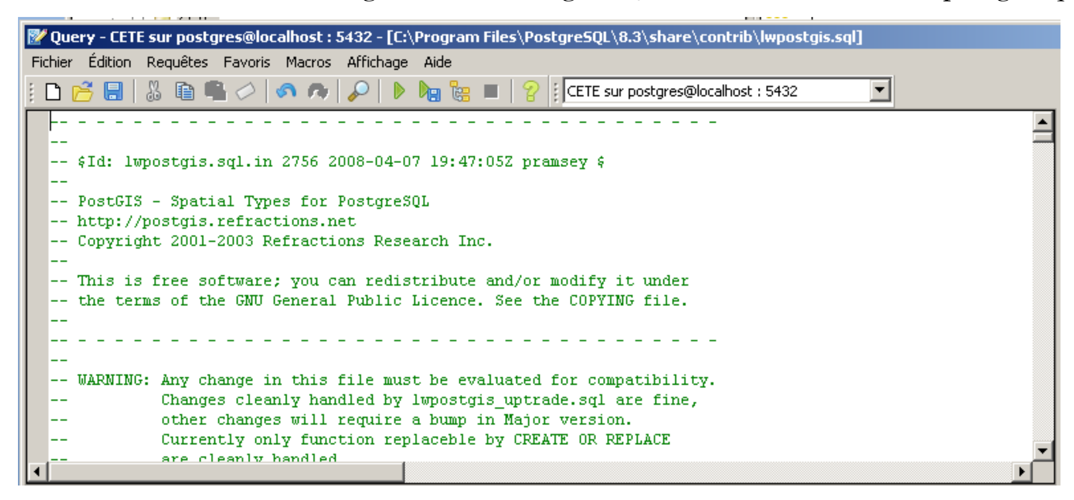
  - Création de la base de données :



- Ajouter la composante spatiale à la base de données (fait conformément à http://www.bostongis.com/PrinterFriendly.aspx?content_name=postgis_tut01)



- Dans la fenêtre d'édition des requêtes, créer le language plpgsql : « **create language plpgsql** »
- Cliquer la flèche verte
- Ouvrir le fichier C:\Program Files\PostgreSQL\8.3\share\contrib\lwpostgis.sql



- Cliquer la flèche verte
- Ouvrir le fichier C:\Program Files\PostgreSQL\8.3\share\contrib\spatial_ref_sys.sql
  A la fin de fichier sql, supprimer la ligne VACUUM ANALYZE spatial_ref_sys;
  Cliquer la flèche verte

Pour permettre l'utilisation de Spit plugin et pour autoriser l'utilisateur « cete » de créer la géométrie des tables stopearea : GRANT ALL ON geometry_columns TO PUBLIC;

D. Déclaration des fonctions plpgsql dans la base

Les fonctions plpgsql sont disponibles dans le répertoire de livraison RequetesSQL.

Elles sont écrites pour l'utilisateur cete ; Dans le cas où la base est utilisée par un autre utilisateur, éditer les scripts et corriger le nom de l'utilisateur cete en le nom souhaité.

Dans une fenêtre de commande, se placer dans le répertoire de livraison RequetesSQL :

```
/cygdrive/e/CETE/RequetesSQL/SQL

Laurent Dezou@LAULO /cygdrive/e/CETE/RequetesSQL/SQL
$ psql -U postgres CETE
Password for user postgres:
Welcome to psql 8.3.4, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit

Warning: Console code page (850) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.

CETE=# \i copy_chemins_in_routes.sql
CREATE FUNCTION
CETE=#
```

- Se placer dans l'environnement psql, utilisateur postgres : `psql -U postgres CETE`
- Créer toutes les fonctions utiles en base de données :
  - `\i copy_chemins_in_routes.sql`
    Création de la fonction `plpgsql` copy_chemin.
    Copie des enregistrements de la table "chemin_AUtlseD31" vers la table "route_AUtlseD31".
  - `\i insert_pathlink_mission.sql`
    Création de la fonction `plpgsql` insert_pathlink_mission.
    Cette fonction prend en paramètre un id de mission.
    Elle crée les liens inter arrêts physiques pour la mission.
    Remarque : les lignes créées sont uniques. Autrement dit, une seule ligne est créée lorsque deux arrêts sont reliés par plusieurs missions.
  - `\i stoparea_calc_stats.sql`
    Création de la fonction `plpgsql` stoparea_calc_stats.
    Cette fonction prend en paramètre une date et une plage horaire..
    Elle calcule les statistiques des arrêts physiques (Cf . VIII.A.3.a).
  - `\i pathlink_calc_stats.sql`
    Création de la fonction `plpgsql` pathlink_calc_stats.
    Note : la fonction get_average_speed écrite en PL/Python est également insérée en base de données. Le langage PL/Python étant à l'heure actuelle « Untrusted » pour PostGreSQL, cette fonction ne peut être créée que par le superuser postgres.
    Cette fonction prend en paramètre un identifiant de mission, une date et une plage horaire.
    Elle calcule les statistiques des tronçons inter arrêts physiques (Cf.VIII.A.3.b).
  - `\i insert_mailles.sql`
    Création de la fonction `plpgsql` insert_mailles.

Calcul des mailles à partir de la table route_AUtlseD31 et insertion des mailles dans la table « maille ».

- `\i insert_deadends.sql`
  Création de la fonction `plpgsql` insert_deadends.
  Calcul des impasses de la table route_AUtlseD31

VII. CRÉATION ET INITIALISATION DE LA BASE DE DONNÉES

A. BD Topo dans PostGIS

Les données de la BD Topo fournies par le CETE Méditerranée sont au format natif du logiciel SIG Mapinfo (.tab).

Le logiciel SIG libre QuantumGIS est utilisé pour copier les données BD Topo dans la base PostGIS.

Dans une première étape, les données de la BD Topo sont transformées en format shapefile (fichiers .shp).

Les fichiers shapefile produits sont ensuite exportés dans la base de données PostGIS.

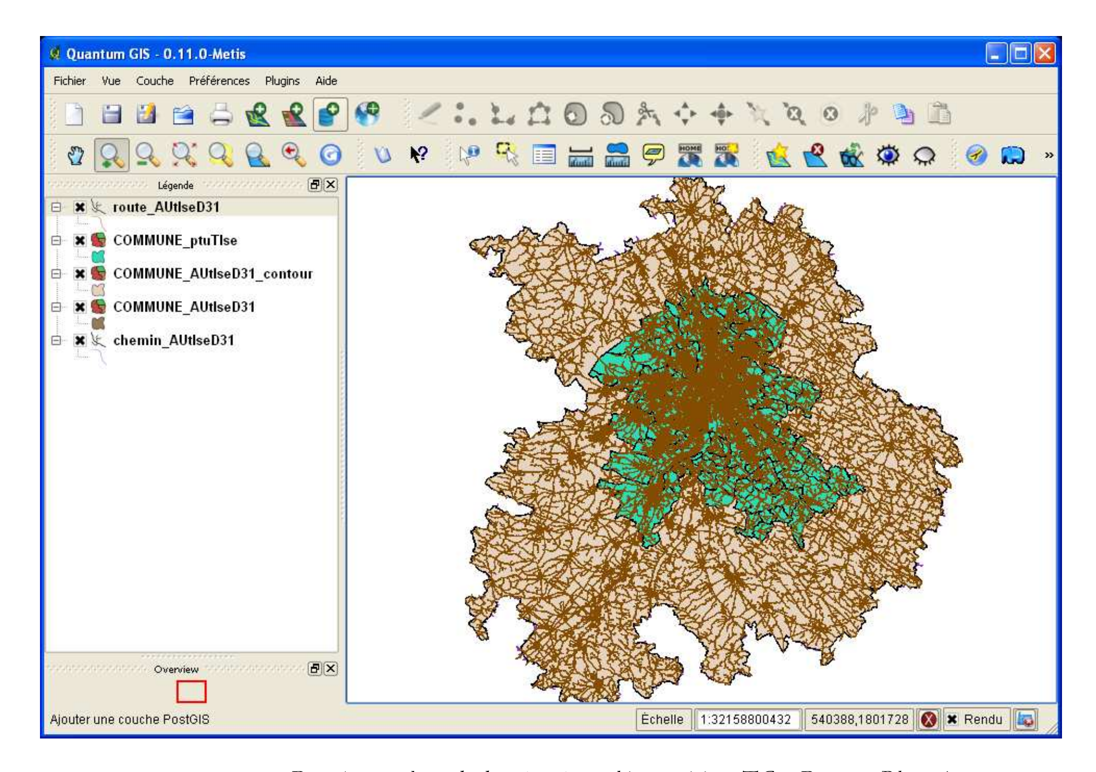
Cas particulier des données route BD Carto : la couche des routes possède les deux champs « Département_Gestionnaire » et « Département ».

Une fois sauvés au format shapefile, ces champs sont tronqués sur 10 caractères en « Départemen » => deux colonnes possèdent les mêmes noms ce qui provoque une erreur lors de l'import dans la base PostGIS.

Nous n'avons pas trouvé comment renommer attribut d'une couche dans QGIS. Pour contourner ce problème, le shapefile a été généré en utilisant le logiciel ArcGIS en version bureautique d'ESRI qui renomme le deuxième champ Départment en « Départem_1 ». Cela aurait aussi possible avec Mapinfo, Géoconcept, ou avec l'utilitaire open source ogr2ogr, en ligne de commande.

1. Transformation des données en shape files

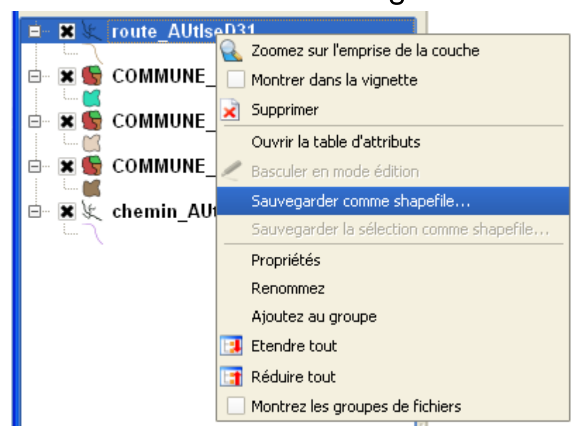
- Dans QuantumGIS, ouvrir les données .tab fournies par le CETE



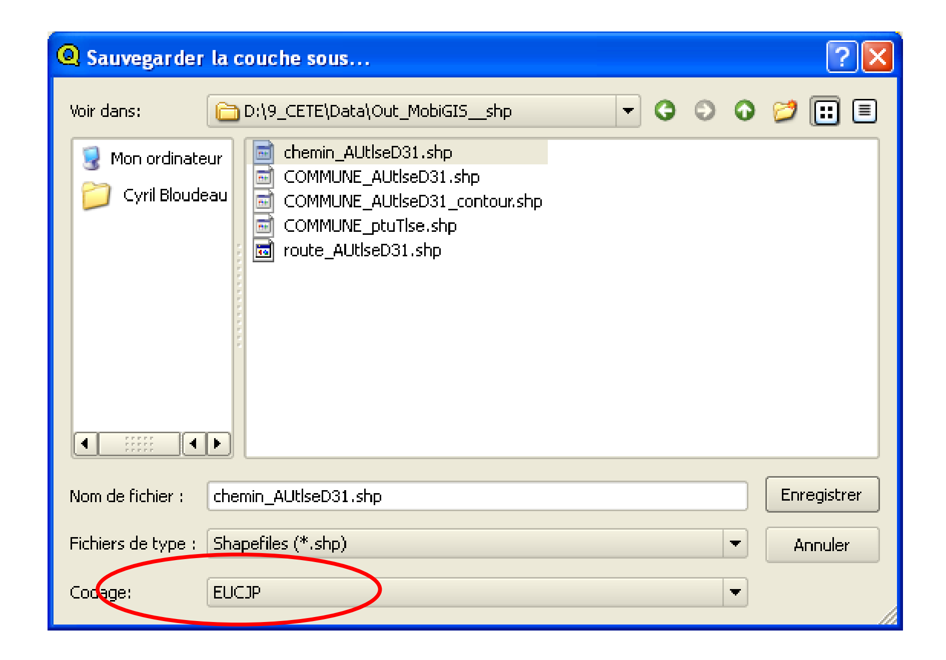




- Clic droit sur une couche et sélectionner « Sauvegarder comme shapefile »



- Sauvegarder le fichier en choisissant son nom, son emplacement et son codage

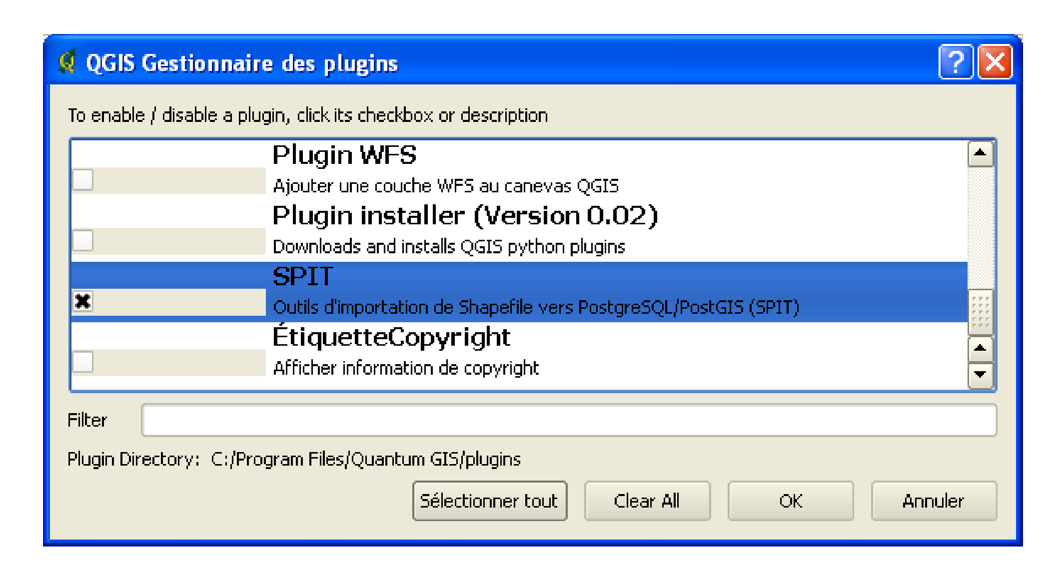


Important : pour permettre une bonne conversion de tous les caractères contenus dans les champs, choisir le **codage : EUCJP**

2. Export des données vers la base PostGIS « CETE »

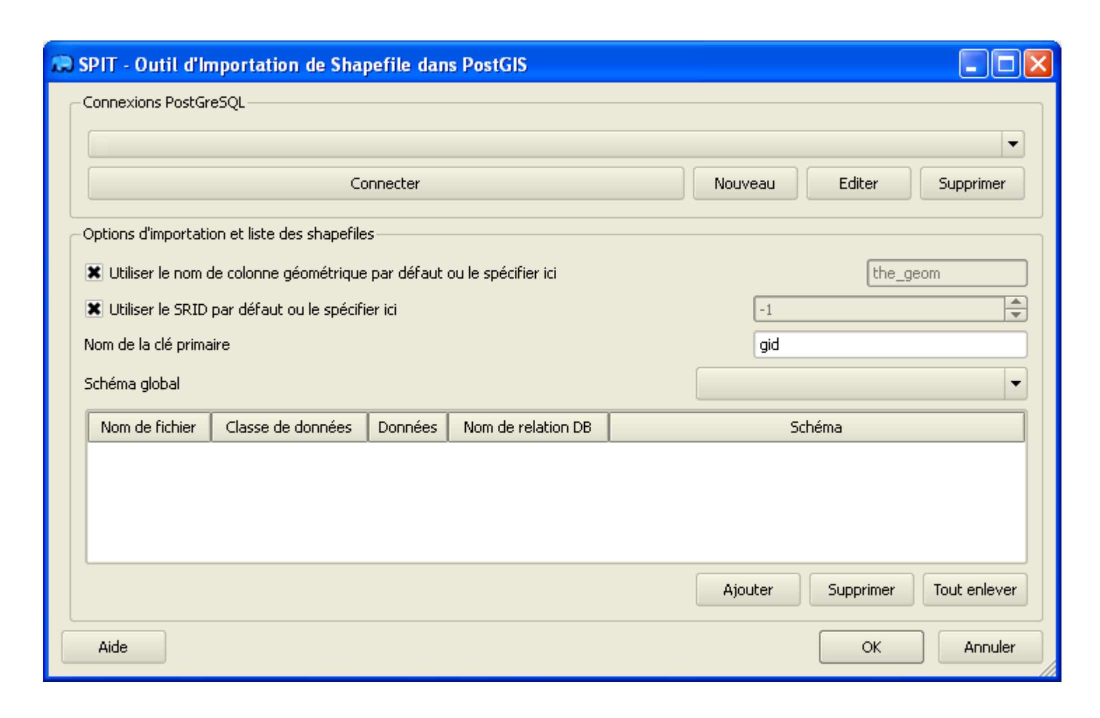
a) Installation du plugin SPIT

Dans la barre de menu de QuantumGIS, cliquer sur Plugins => Gestionnaire de Plugins. Sélectionner l'extension « SPIT » (export vers PostgreSQL/PostGIS)

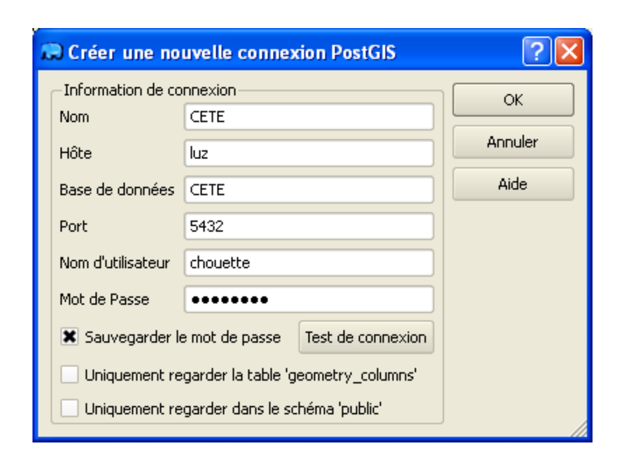


b) Export des données vers PostGIS

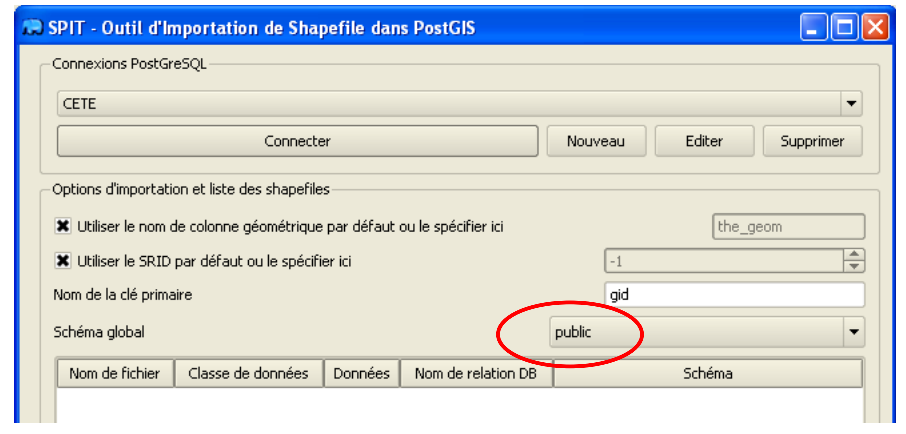
- Cliquer sur l'icône pour ouvrir la fenêtre de paramétrage de l'export vers PostgreSQL/PostGIS



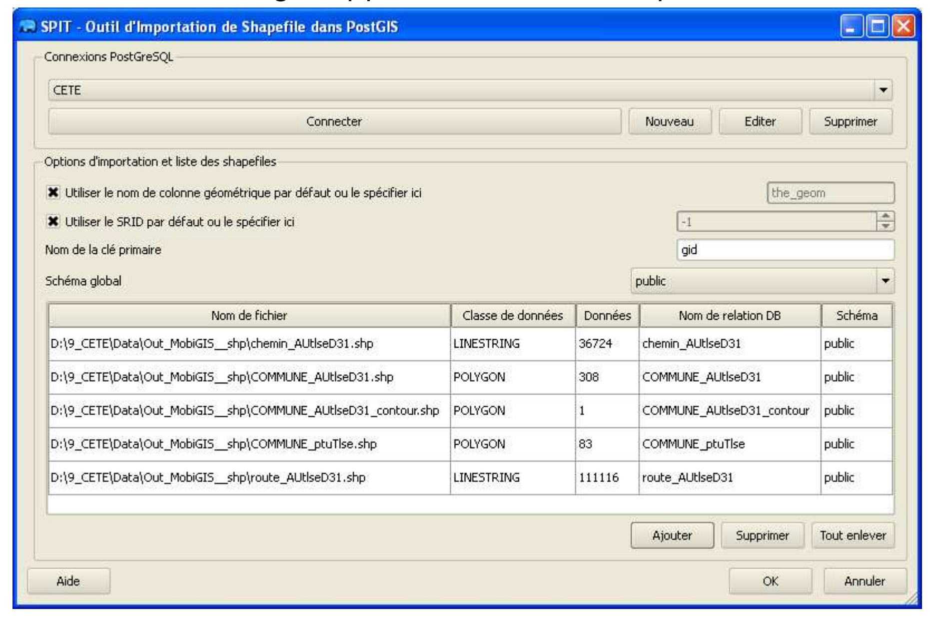
- Cliquer sur Nouveau pour ouvrir la fenêtre de paramétrage de la connexion vers la base « CETE », renseigner les différentes informations, tester la connexion et valider



- En cliquant sur « Connecter », le schéma global dans lequel vont être ajoutées les données apparaît



- Cliquer sur ajouter, sélectionner les shapefiles à ajouter à la base de données et valider. Les données à charger apparaissent. Enfin, cliquez sur Ok.

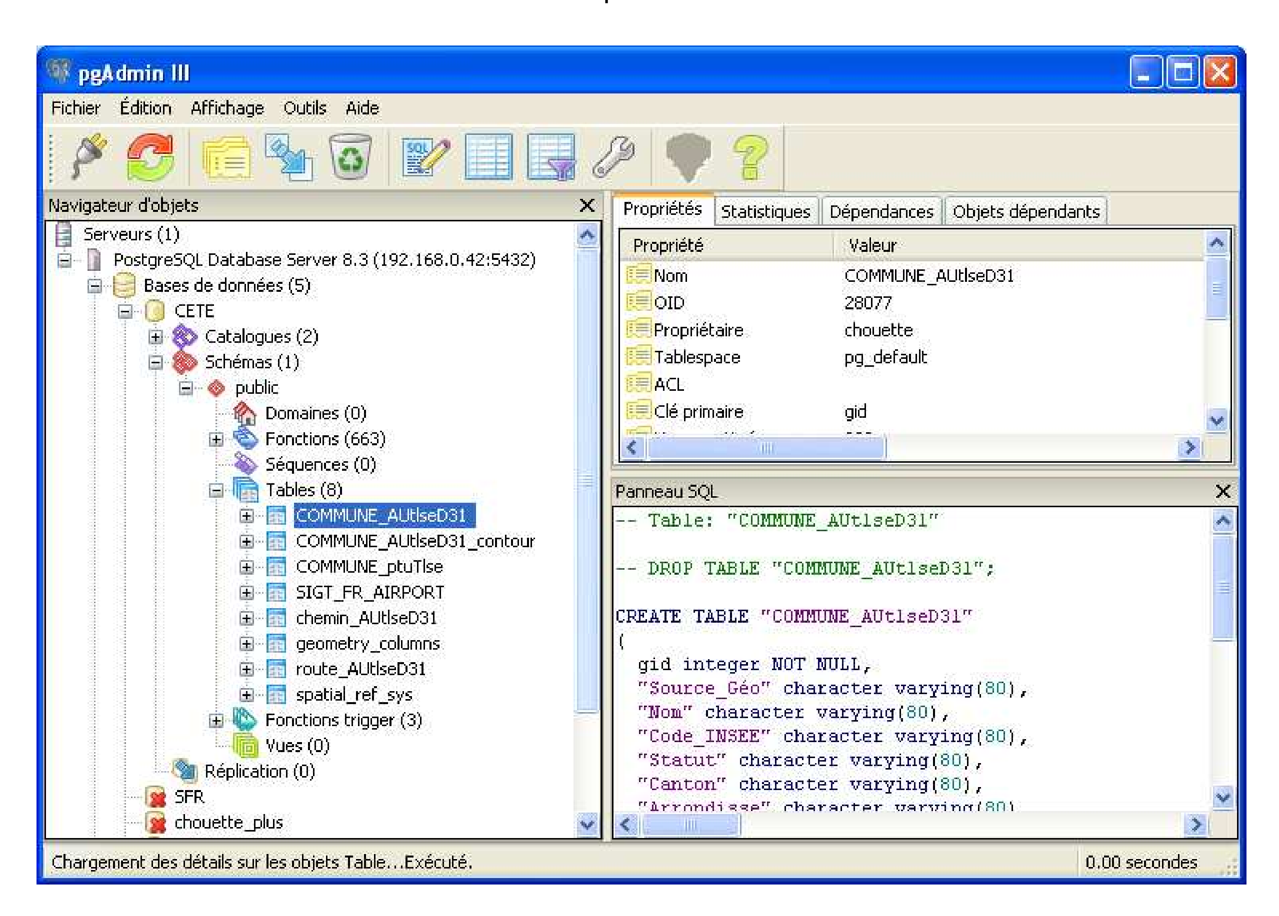






3. Vérifier l'export des données vers la base « CETE »

- Ouvrir PG Admin et sélectionner la base de données « CETE ». Vérifier la présence des tables dans CETE => Schémas => public => tables



4. Copie des chemins dans la table des routes

Dans une fenêtre psql : `select copy-chemin();`
Cette commande est nécessaire car la voirie est décrite dans la base BD Topo de l'IGN par 2 tables (chemins et routes) qu'il faut fusionner en une seule pour pouvoir la traiter en tant que couche. Pour d'autres sources de données voirie disponibles dès le départ dans une seule couche, ce script SQL pourrait être inutile ou à adapter.

Attention, cette commande ne doit être exécutée qu'une fois, sous peine de voir les chemins dupliqués dans la table des routes.

5. Ajout de la colonne impasse à la table route_AUtlseD31

Dans une fenêtre psql : `ALTER TABLE "route_AUtlseD31" ADD COLUMN deadendflag boolean DEFAULT false;`

Cette commande se trouve également dans le script `create_columns_routes.sql` livré, qui ajoute également un index (routegeom_idx).

B. Données TC Chouette

Création de la structure de la base de données CHOUETTE.

Les données CHOUETTE peuvent être insérées en base à partir d'un dump PostgreSQL ou

directement à l'aide de l'outil CHOUETTE.

1. Insertion des données CHOUETTE à partir d'un dump database

Cette partie sera complétée dans le rapport phase 2.

2. Insertion des données CHOUETTE à l'aide de l'outil CHOUETTE

Cette partie sera complétée dans le rapport phase 2.

VIII. Requêtes dans la base PostgreSQL / PostGIS

A. Requêtes dans la base de données TC

1. Ajout de la structure géométrie aux tables CHOUETTE

Note : la géométrie est intégrée aux versions de CHOUETTE supérieures à la version 1.3 : Les requêtes de création des géométries décrites dans ce chapitre sont alors inutiles.

De plus, dans ces versions, un trigger met automatiquement à jour la géométrie des éléments lorsque les champs latitude ou longitude sont modifiés dans la base de données PostgreSQL, notamment depuis l'application web.

a) Table `stoparea`

Le script de création de la géométrie des points d'arrêt est create_geo_stoparea.sql, il contient le code SQL suivant :

```
ALTER TABLE stoparea ADD COLUMN gid serial NOT NULL;
ALTER TABLE stoparea ADD CONSTRAINT unique_gid UNIQUE (gid);

-- Création de la colonne géométrique, EPSG:4326 = WGS84
SELECT AddGeometryColumn('', 'stoparea','geom',4326,'POINT',2);

CREATE INDEX stopareageom_idx ON stoparea
   USING GIST ( geom GIST_GEOMETRY_OPS );
   -- Création de la colonne journeynb, nombre de courses passant sur une
   période donnée
ALTER TABLE stoparea ADD COLUMN journeynb integer NOT NULL DEFAULT 0 ;
```

Ce script doit être exécuté une et une seule fois, après que la structure de la base de données CHOUETTE ait été créée. Il peut être exécuté dans une fenêtre de commande psql, connecté en tant qu'utilisateur cete :

```
-- \i create_geo_stoparea.sql ;
```

b) Table `pathlink` :

La table pathlink contient des traits rectilignes directs (type « vol d'oiseau ») qui relient deux arrêts physiques (items de la table stoparea) successifs d'au moins une mission.

Deux scripts sont livrés :

- `create_pathlink.sql` : script de création de la table pathlink dans la base de données CHOUETTE

Utilisation :

- Création de la table pathlink.
  Dans une fenêtre psql : `\i create_pathlink.sql ;`

2. Mise à jour des géométries

a) Table stoparea :

```
-- Maj de la colonne sur la base des longitudes/latitudes
UPDATE stoparea SET geom = PointFromText('POINT(' || longitude || ' ' ||
latitude || ')',4326) ;
```

b) Table pathlink :

```
truncate table pathlink ;

select insert_pathlink_mission(t.id) from journeypattern t order by t.id;
```

Exemple de visualisation dans QGIS :

3. Calcul des indicateurs

Les indicateurs sont calculés sur une période constituée d'une date et d'une plage horaire : date et horaires sont des paramètres d'entrée de la procédure plsql en charge du calcul des indicateurs.

a) Table `stoparea` :

Un indicateur est calculé pour chaque point d'arrêt :

- **journeynb** : nombre de courses passant par le point d'arrêt sur la période

Utilisation :

- Calcul des statistiques pour tous les points d'arrêt :

```
Update stoparea set journeynb = 0 ;
select stoparea_calc_stats('2008-04-02', '05:20:05', '10:20:05') ;
```

Exemple de visualisation dans QGIS pour une requête `select stoparea_calc_stats('2008-11-14', '05:20:05', '20:20:05') ;` :

Remarque : couleurs et plages de valeurs sont définies automatiquement par QGIS pour un nombre de classe de valeurs choisi par l'utilisateur. Il est possible alors, de changer les plages de valeur, de changer les couleurs ou encore, on pourrait choisir un symbole de taille fonction de la fréquence de passage à l'arrêt.
Pour ce dernier point, dans ArcMap d'ArcGIS par exemple et non dans GQIS, un symbole de taille proportionnelle à la fréquence de passage à l'arrêt pourrait être défini.

b) Table `pathlink` :

Les indicateurs sont calculés sur une période constituée d'une date et d'une plage horaire : date et horaires sont des paramètres d'entrée de la procédure plsql en charge du calcul des indicateurs.

Indicateurs calculés pour chaque tronçon:

- **avgspeed** : vitesse moyenne sur le tronçon.

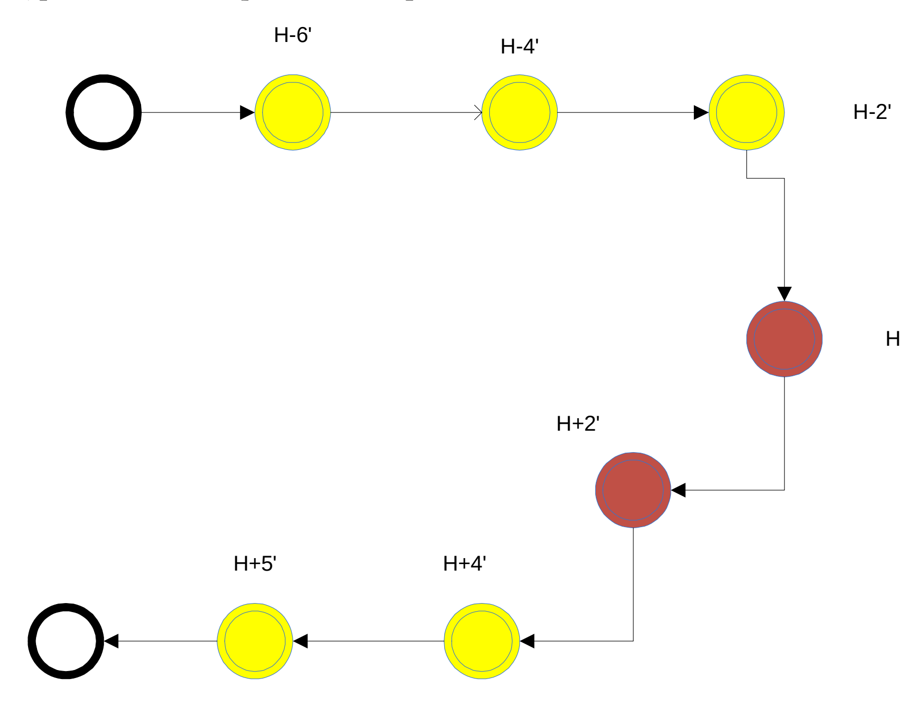
  La moyenne est calculée sur l'ensemble des courses empruntant le tronçon (pathlink) considéré, sur la plage horaire étudiée.

  Dans les faits, les horaires indiqués entre deux points d'arrêts consécutifs sont non significatifs (parfois 0 seconde !) ou trop approximatifs (l'unité est la minute), car les horaires théoriques sont renseignés sur une longueur de trajet significative qui regroupe plusieurs points d'arrêts. Par conséquent, les horaires ne peuvent être utilisés en l'état pour calculer les vitesses moyennes : il est nécessaire d'élargir le calcul en intégrant les tronçons voisins si on veut obtenir une vitesse moyenne significative.

  Dans notre script SQL, la moyenne est calculée sur une « fenêtre de temps » de 10 minutes autour du tronçon à l'étude ; il est en effet estimé qu'à partir de 10 minutes de temps de trajet, la marge d'erreur liée aux horaires n'influe pas de manière majeure sur les résultats. Ce calcul sur une fenêtre de temps implique que, selon les courses, un nombre variable de tronçons autour du tronçon étudié sont pris en compte.

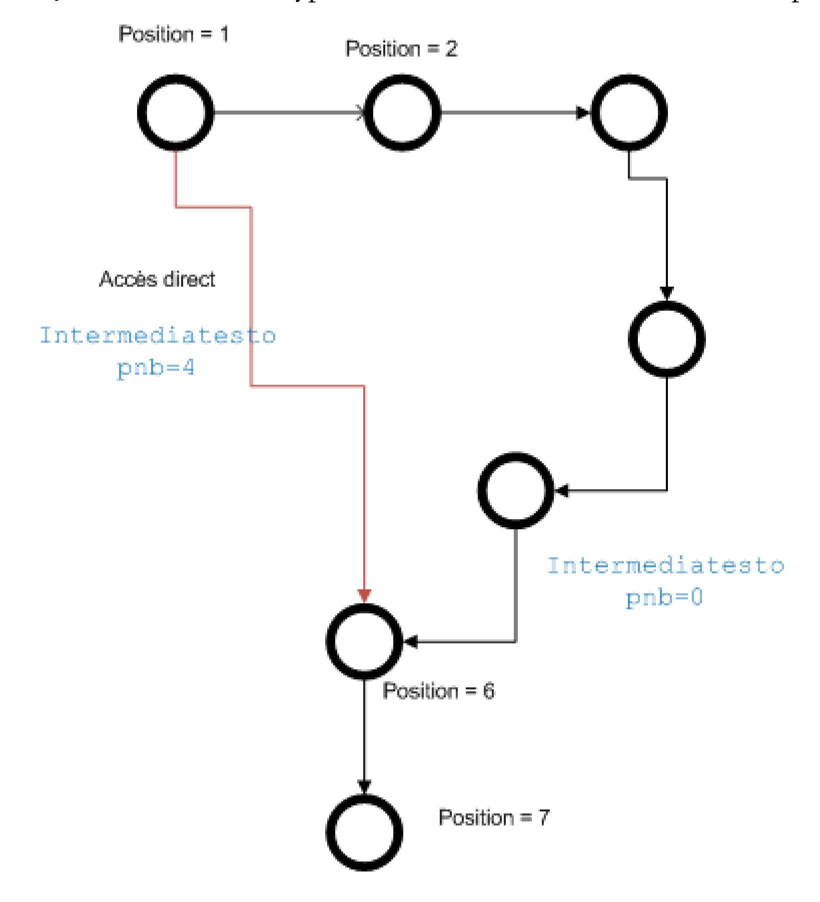
  Les moyennes sont calculées à partir de courses passant par le tronçon mais qui n'appartiennent pas forcément toutes à la même mission, et donc s'arrêtant à des arrêts différents selon la mission… cette notion est prise en compte par l'attribut intermediatestopnb de pathlink.

  La vitesse moyenne est calculée sur une zone d'au moins 10 minutes autour du tronçon.
  Dans l'exemple ci-dessous, pour calculer la moyenne sur le tronçon H -> H+2 (en rouge) les 7 points d'arrêt colorés en jaune sont pris en compte, la distance est égale à la somme des lignes type vol d'oiseau qui relient les points d'arrêt.



L'ensemble des courses de la période est parcouru : avgspeed est la moyenne des vitesses moyennes obtenues pour chaque course passant par le tronçon durant la période.

- **linknb** : nombre de courses reliant les deux points d'arrêt dans la période. (de manière directe ou non, pour toutes les lignes du réseau décrites en base de données)
- **intermediatestopnb** : permet d'identifier les tronçons d'une course de type « direct ».
  En effet, parmi les missions considérées, certaines empruntent des voies directes. Le champ intermediatestopnb contient le nombre d'arrêts évités par un tronçon direct de ce type de mission comparé à la mission de type « omnibus ».
  Par exemple, le tronçon ci-dessous où l'on voit 4 arrêts évités par la mission directe se voit attribué un intermediatestopnb de 4.
  Tous les tronçons inter-arrêt de type « Omnibus » ont un intermediatestopnb qui vaut 0.



Utilisation :

- Calcul des statistiques pour tous les tronçons :

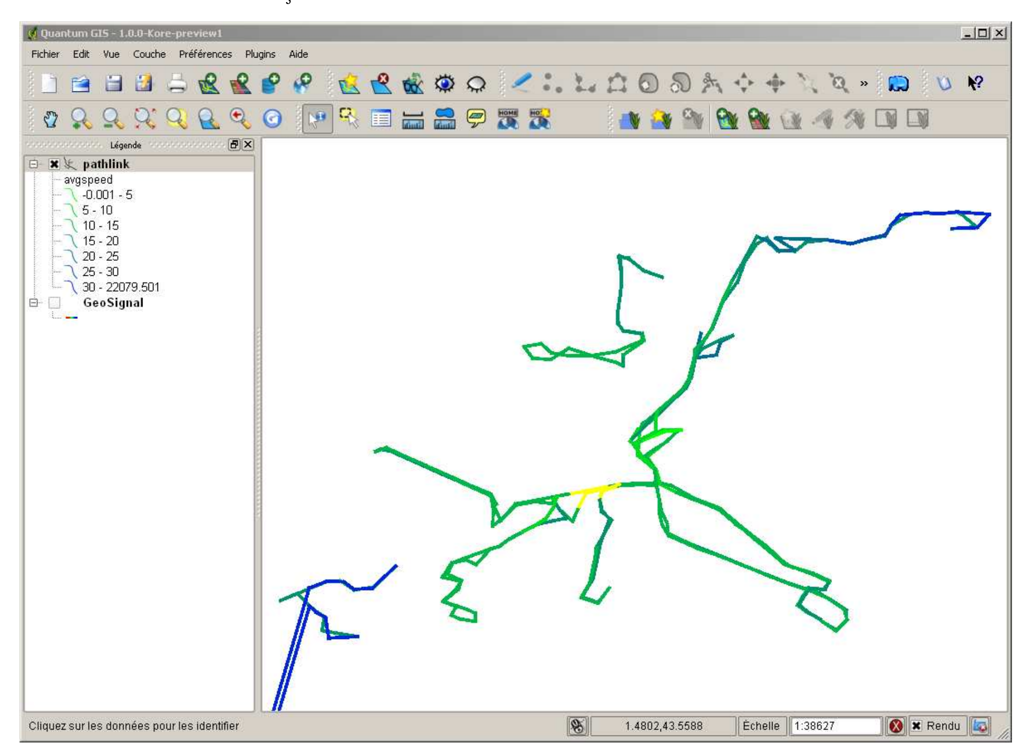
```
Update pathlink set (linknb, intermediatestopnb, avgspeed) = (0, 0, 0) ;
select pathlink_calc_stats(t.id, '2008-04-02', '05:20:05', '10:20:05') from journeypattern t;
```





Exemple de visualisation dans QGIS pour une requête `select pathlink_calc_stats(t.id,'2008-11-14', '05:20:05', '20:20:05') from journeypattern t;`

Colorisation des tronçons selon la vitesse :



Colorisation des tronçons selon le nombre de courses reliant les deux points d'arrêt :

Quantum GIS - 1.0.0-Kore-preview1
Fichier Edit Vue Couche Préférences Plugins Aide
Légende
pathlink
linknb
-0.001 - 39.143
39.143 - 78.286
78.286 - 117.429
117.429 - 156.571
156.571 - 195.714
195.714 - 234.857
234.857 - 274.001
GeoSignal
Zoom to Selection
1.4667,43.5713
Échelle
1:38627
Rendu




Remarque : comme pour la colorisation des points d'arrêt, les couleurs et plages de valeurs sont définies automatiquement par QGIS pour un nombre de classe de valeurs choisi par l'utilisateur. Il est possible alors, de changer les plages de valeur, de changer les couleurs.
Dans ArcMap d'ArcGIS par exemple, une palette de couleurs peut être choisie pour afficher une graduation complète de couleurs en fonction de la valeur de l'indicateur.

B. Requêtes dans la base de données voirie

1. Copie des données chemin dans la table des routes

Les chemins et les routes sont contenus dans deux tables distinctes extraites de la BD TOPO IGN pour l'aire urbaine de Toulouse, respectivement "chemin_AUtlseD31" et "route_AUtlseD31". Les chemins sont copiés dans la table des routes qui contiendra l'ensemble des données voirie utilisées. Cette requêtes a déjà été présentée plus haut (VII.A.4) et ne doit pas être exécutée deux fois.

Pour cela, dans une fenêtre psql : `select copy_chemin();`

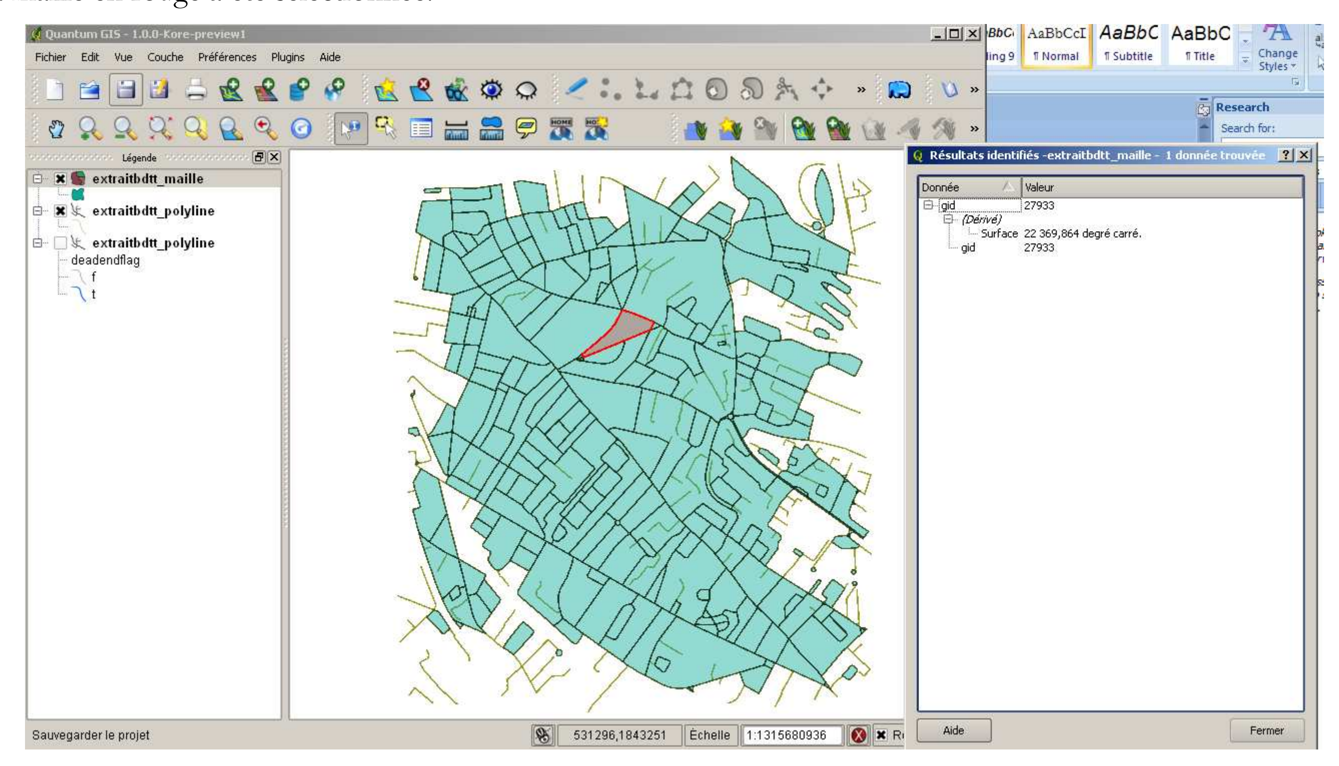
2. Recherche des mailles

Une maille est une liste chaînée d'arcs en boucle, qui part et aboutit au même noeud (un « chemin élémentaire » dans le vocabulaire de la théorie des graphes).

La recherche des mailles est basée sur la fonction native de PostGIS ST_Polygonize : « *Creates a GeometryCollection containing possible polygons formed from the constituent linework of a set of geometries* ».

A noter que le calcul des mailles est assez long, environ 1h30 pour la BD Topo de l'aire toulousaine sur un PC XP, Intel Core 2 Duo T7500 2,20 GHz avec 2GO de RAM.

La copie d'écran ci-dessous présente le résultat de la recherche de mailles pour un extrait de BD Topo. Les polygones constitués apparaissent en noir, les routes ne faisant partie d'aucune maille sont visibles en couleur verdâtre.
La maille en rouge a été sélectionnée.

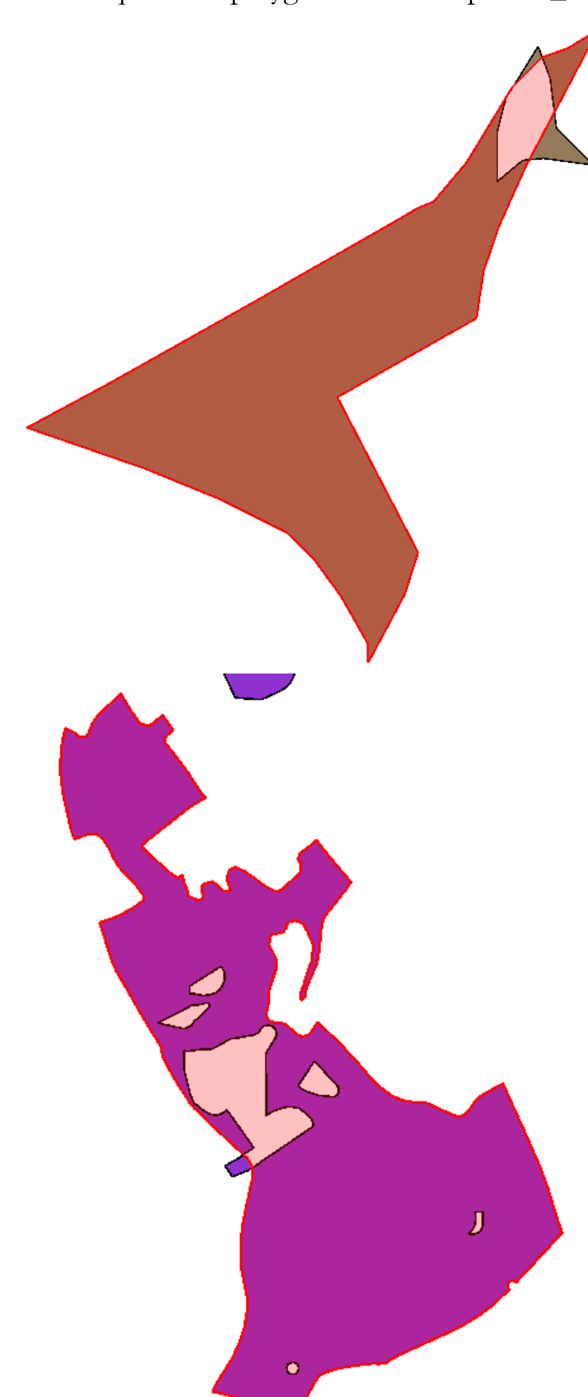


Il faut commencer par créer la table MAILLE (script `create_mailles.sql`).

Pour lancer le calcul des mailles et peupler la table mailles, dans une fenêtre psql : `select`

```
insert_mailles();
```

A noter que deux polygones calculés par ST_Polygonize ont des géométries non valides :





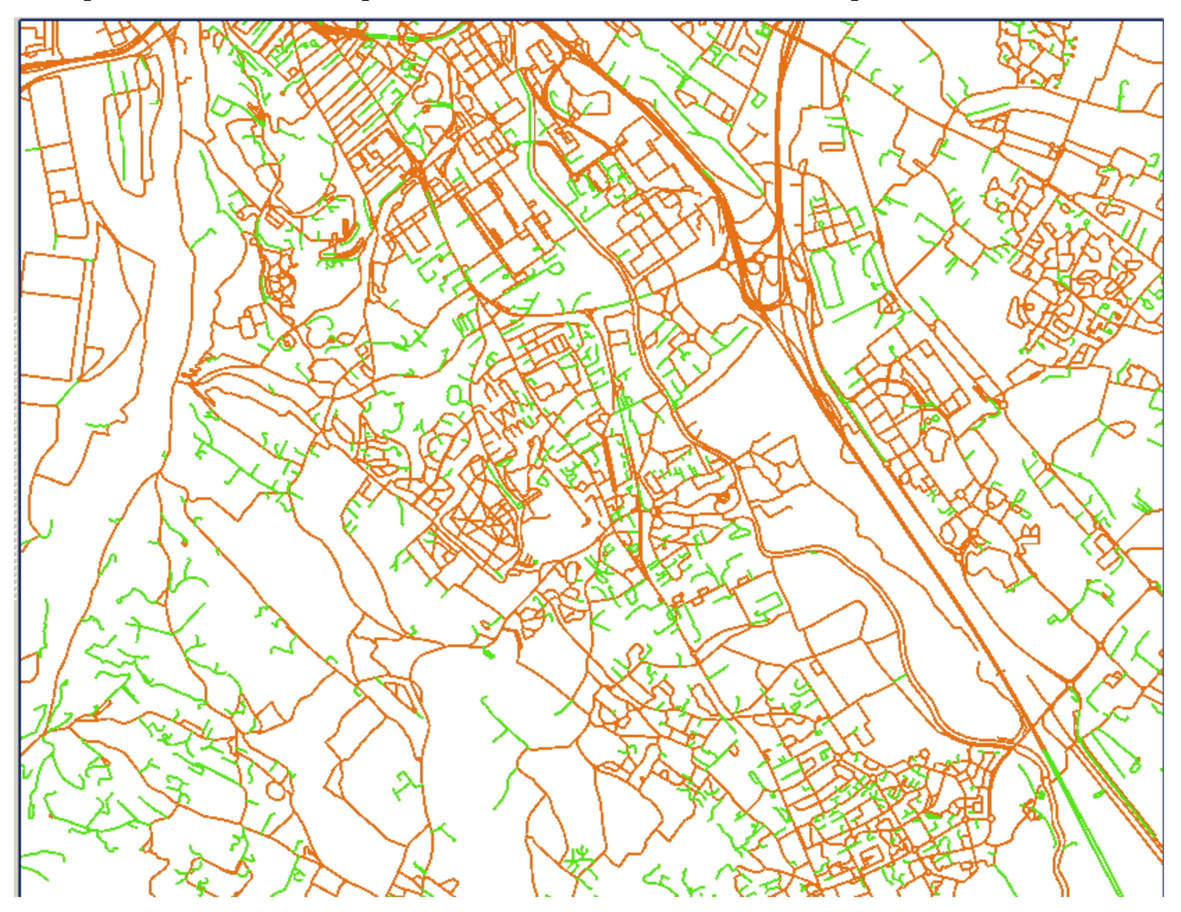

3. Recherche des impasses

Préalable au calcul : les mailles doivent avoir été calculées.

Les arcs entièrement contenus dans les mailles sont recherchés, ils sont marqués comme étant des impasses.

Le calcul des impasses a pris 400 s sur le serveur de données pour l'aire urbaine toulousaine.

La copie d'écran ci-dessous présente le résultat de la recherche d'impasses dans QGIS /



Pour lancer le calcul des impasses et renseigner le champ deadendflag de la table "route_AUtlseD31", dans une fenêtre psql :

```
select insert_deadends();
```

4. Calcul des indicateurs voirie

a) Taux d'impasses

Dans une fenêtre psql :

```
select count(*) from "route_AUtlseD31" where deadendflag=true;
select count(*) from "route_AUtlseD31" where deadendflag=false;
```

Le nombre d'impasses sur les données de l'agglomération toulousaine est de 37230 **soit 25,1 %.**

b) Distribution de la taille des mailles

La colonne « surface » contient la donnée surface de chaque maille. Le calcul de la surface est réalisé





en SQL dans une fenêtre psql :

```
update maille set surface=ST_AREA(geom) ;
```

Un affichage gradué des surfaces peut être réalisé dans QGIS :

Pour obtenir un fichier de résultats interprétable dans Excel. Dans une fenêtre psql :

`\o results.txt` -- Les résultats de la commande sont écrits dans le fichier results.txt du répertoire courant

```
select surface, gid from maille;
```

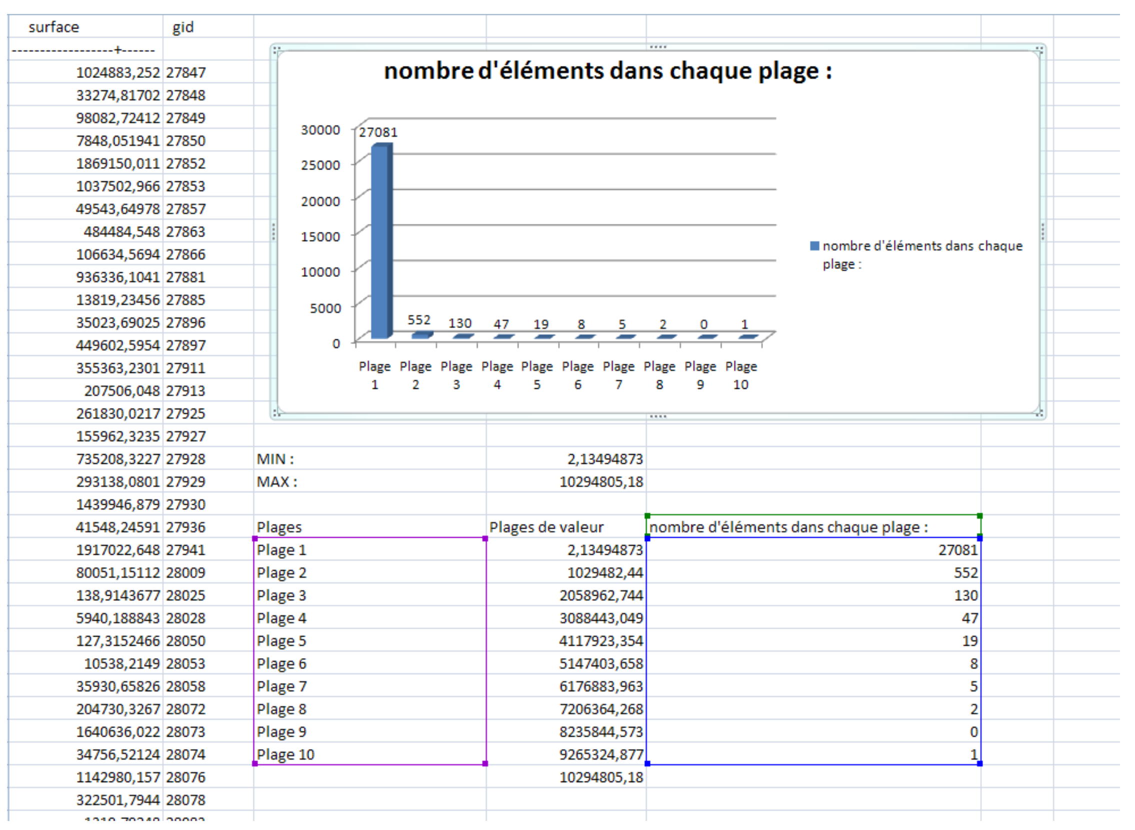
Les données sont contenues dans le fichier results.txt, elles peuvent être interprétées par un outil du type Excel par exemple :

- Ouvrir le fichier results.txt dans Excel en utilisant l'option Delimited, choisir le caractère | comme délimiteur

- Des exemples des graphes sont fournis ci-après :
  - Nombre d'éléments par plage de valeurs





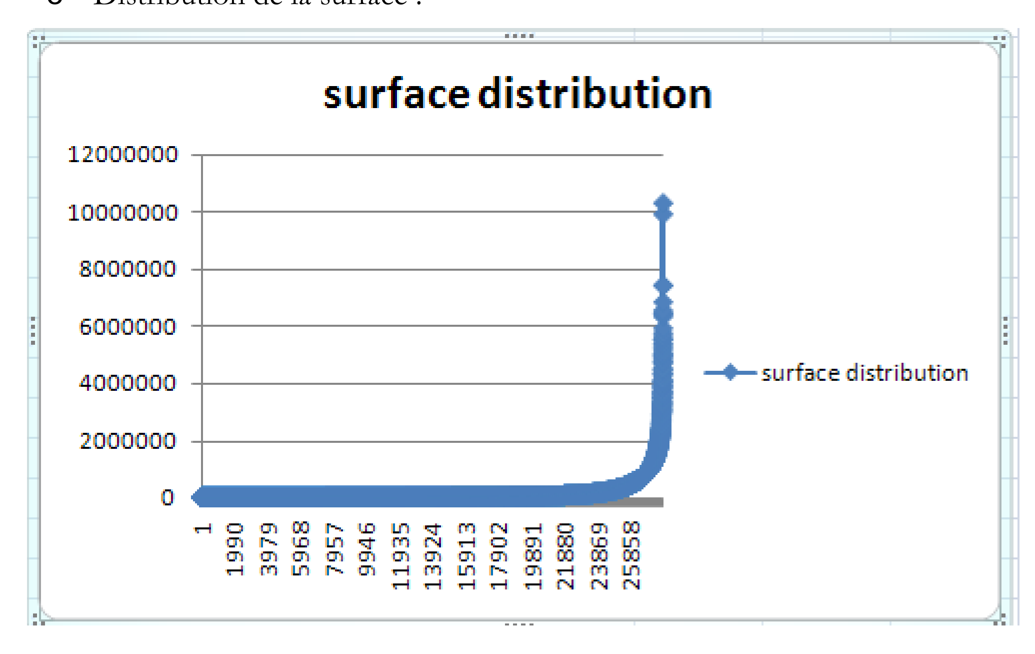


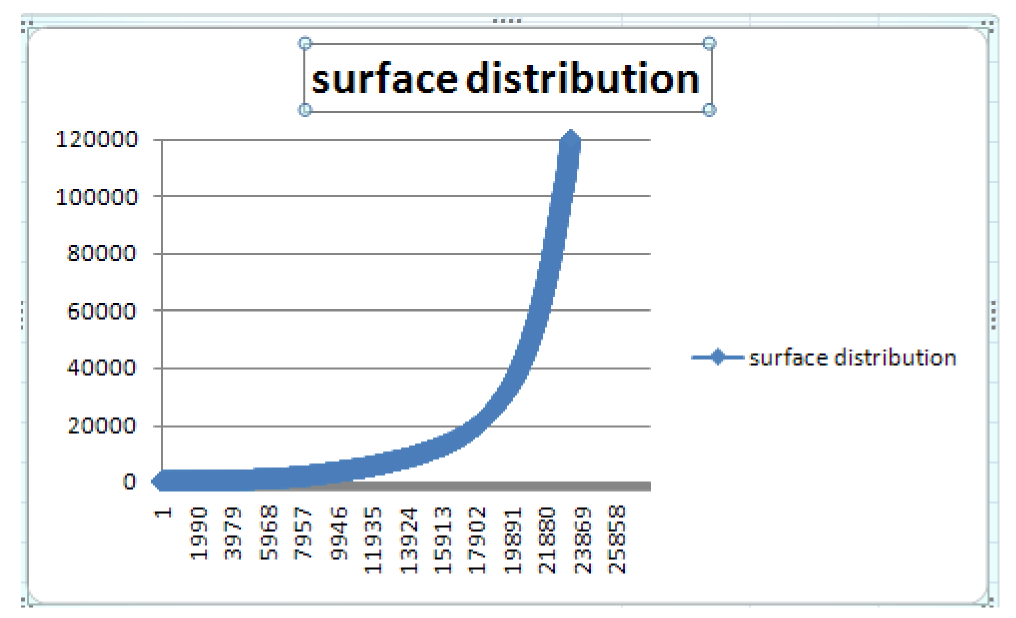
surface	gid
1024883,252	27847
33274,81702	27848
98082,72412	27849
7848,051941	27850
1869150,011	27852
1037502,966	27853
49543,64978	27857
484484,548	27863
106634,5694	27866
936336,1041	27881
13819,23456	27885
35023,69025	27896
449602,5954	27897
355363,2301	27911
207506,048	27913
261830,0217	27925
155962,3235	27927
735208,3227	27928
293138,0801	27929
1439946,879	27930
41548,24591	27936
1917022,648	27941
80051,15112	28009
138,9143677	28025
5940,188843	28028
127,3152466	28050
10538,2149	28053
35930,65826	28058
204730,3267	28072
1640636,022	28073
34756,52124	28074
1142980,157	28076
322501,7944	28078

MIN :	2,13494873
MAX :	10294805,18

Plages	Plages de valeur	nombre d'éléments dans chaque plage :
Plage 1	2,13494873	27081
Plage 2	1029482,44	552
Plage 3	2058962,744	130
Plage 4	3088443,049	47
Plage 5	4117923,354	19
Plage 6	5147403,658	8
Plage 7	6176883,963	5
Plage 8	7206364,268	2
Plage 9	8235844,573	0
Plage 10	9265324,877	1
	10294805,18	

- Distribution de la surface :



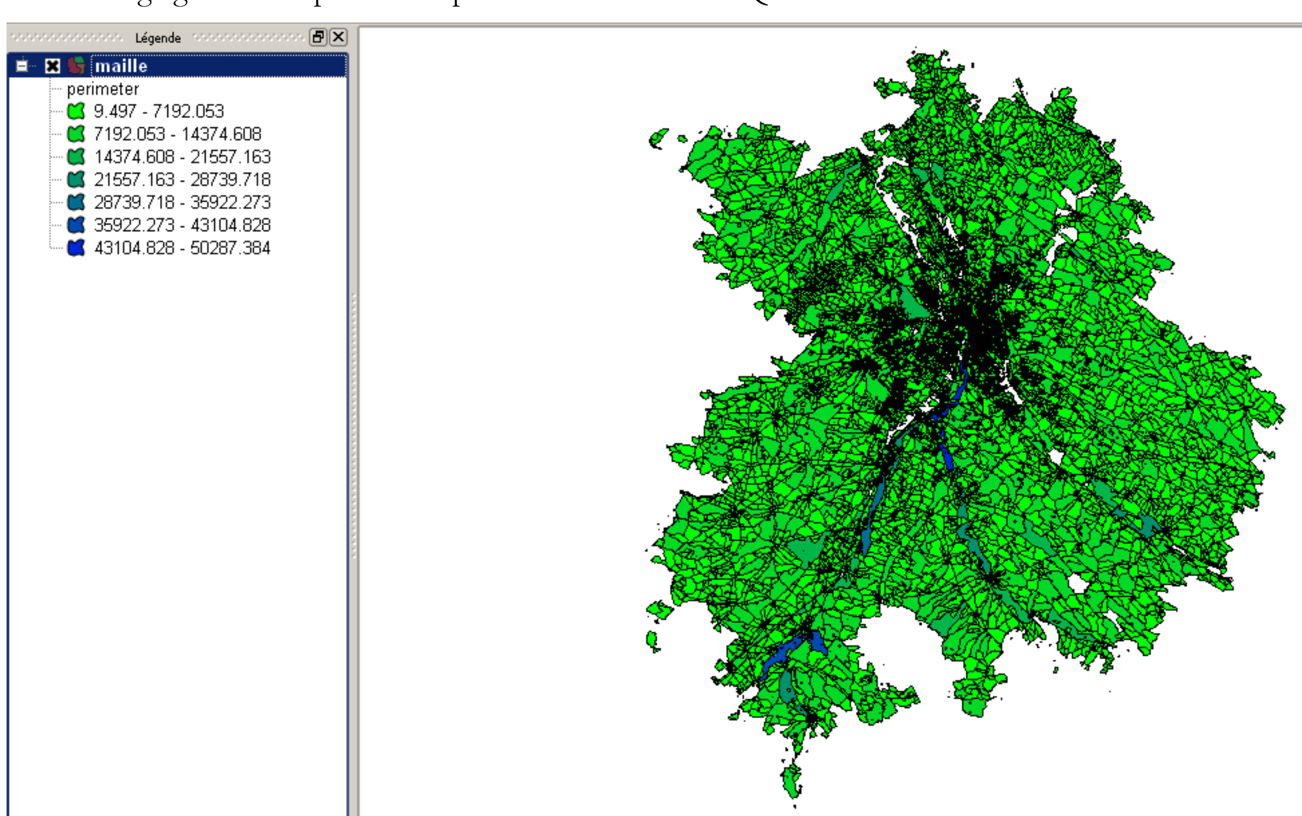


c) <u>Valeurs Min, Max, moyenne et médiane du périmètre des mailles</u>

La colonne « perimeter » contient la donnée périmètre de chaque maille. Le calcul de périmètre est réalisé en SQL dans une fenêtre psql :

```
update maille set perimeter=ST_PERIMETER(geom) ;
```

Un affichage gradué des périmètres peut être réalisé dans GQIS :



Calcul des indicateurs :

- Valeur Min : `select min(perimeter) from maille;`

- Valeur Max : `select max(perimeter) from maille;`
- Valeur médiane (attention, le calcul est un peu long, il a pris 574 s sur un serveur Intel XEON 4 cœurs) :

```
CREATE OR REPLACE FUNCTION occurrences_cumulees(FLOAT) RETURNS INTEGER
AS
      'SELECT sum(1)::INTEGER FROM maille where perimeter <= $1;'
 LANGUAGE SQL;

CREATE OR REPLACE FUNCTION Moitie_population() RETURNS INTEGER AS
      'SELECT count(*)::INTEGER/2 FROM maille;'
 LANGUAGE SQL;

SELECT min(perimeter) AS median FROM maille
   WHERE occurrences_cumulees(perimeter) >= Moitie_population();
```

- Valeur moyenne : `select avg(perimeter) from maille;`

IX. PERSPECTIVES ET SUITE DU TRAVAIL

Ce chapitre sera complété lors de la phase 2 de consolidation qui va consister à :

- Rédiger une introduction décrivant le contexte et les objectifs, afin de cibler des utilisateurs potentiels
- Améliorer le package du démonstrateur : l'objectif est de faciliter son installation et permettre la réalisation de démonstrations,
- Pour la partie TC, tester si possible sur des données Tisséo récentes. La prise en compte de ces données sera documentée,
- Mener une réflexion sur les indicateurs calculés : quels indicateurs ont le plus de sens ? Peut-on définir de nouveaux indicateurs utiles ? etc.
- Améliorer les traitements des données de la base PostGIS : quels rapports peut-on définir ? Cartes thématiques pouvant être générées dans un SIG libre (QGIS, ou autre) ?

Par ailleurs, les résultats de cette étude pourront être intégrés dans le cadre du projet POTIMART: notamment les modèles des données POTIMART et celui de l'étude sont communs, la géolocalisation des points d'arrêt et des tronçons TC pourra être utilisée sans modification dans POTIMART.

FIN DU DOCUMENT